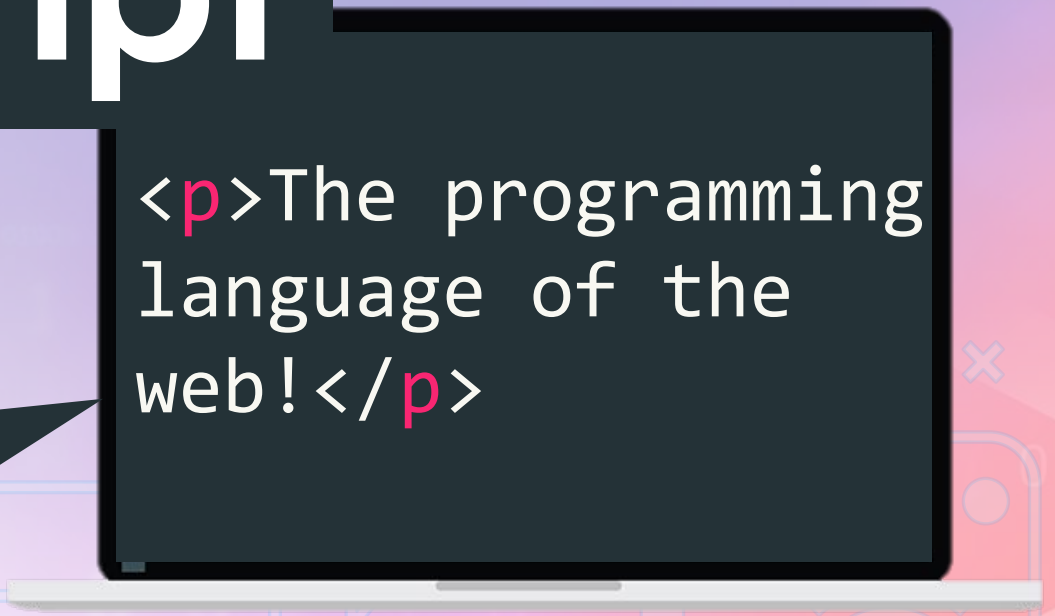
Sentinel

<Delve into Web Dev>

DEFENDING OUR DIGITAL WAY OF LIFE

JavaScript



Powers most of the **dynamic behavior**

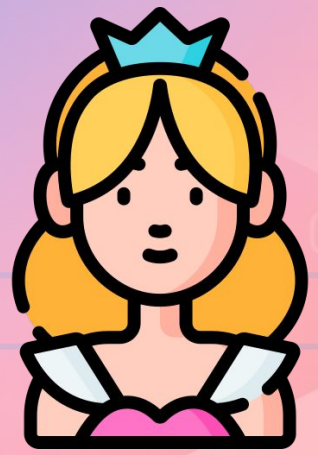
Recap

Element	Function
	Block Container Element
	Inline Container Element
<textarea>	Create text area that can hold unlimited characters
<table>	Create Table

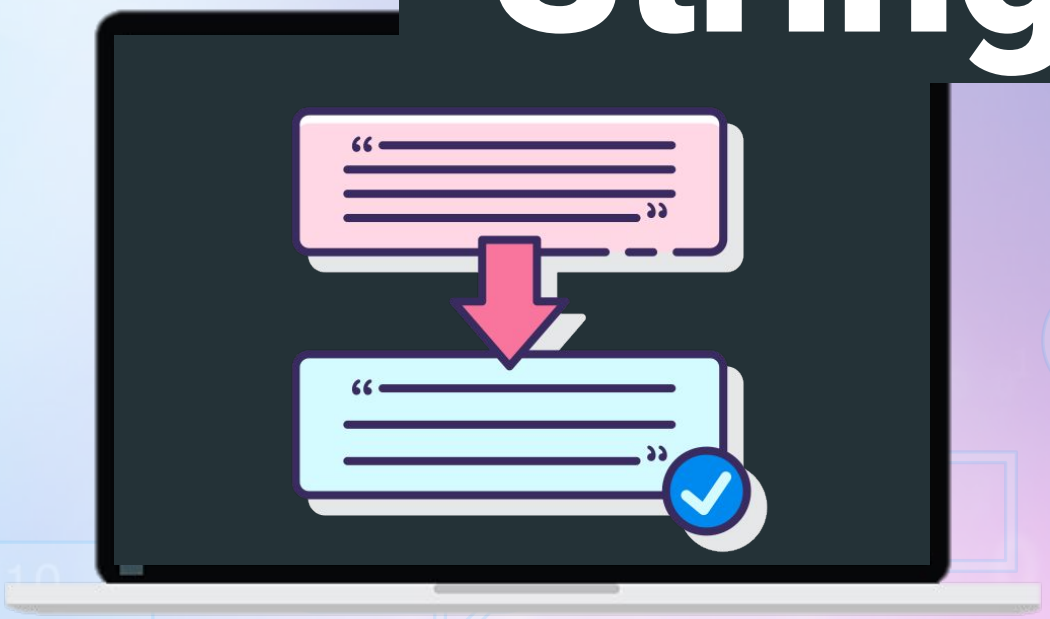
Recap

```
<table>
    <thead>...</thead>
    <tbody>
        <tr>
            <td>Princess</td>
            <td>Carolyn</td>
        </tr>
        <tr>...</tr>
    </tbody>
</table>
```

First Name	Last Name
Princess	Carolyn



“Strings Cont’d”



Learning Objectives

- To represent a new-line in our strings
- To get the char at a particular index
- To get the length of a string
- Convert a string to uppercase and lowercase
- Replace occurrences of a substring
- Slice a string to return the chars between a specific pair of indexes

Loooong Strings

Sometimes we need our strings to span multiple lines

- let myLongString = "*Lets suppose that you were able" +*
- *"every night to dream any dream you wanted to dream" +*

```
let myLongString = "Lets suppose that you were able" +
 "every night to dream any dream you wanted to dream" +
"And you would naturally as you began on this" +
 "adventure of dreams, you would fulfill all your wishes"
```

New Line

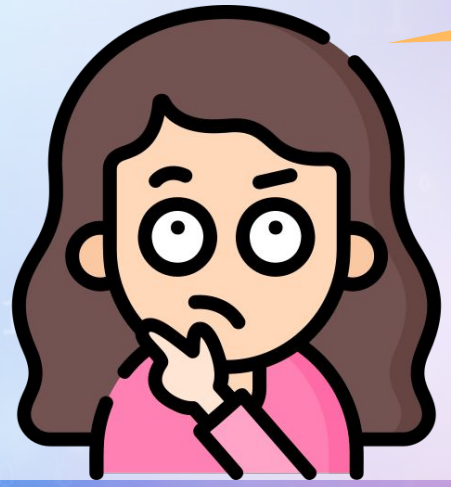
But then, how would we represent an actual new-line in our string?

\n

```
let myString = "You are not a drop in the ocean.\n" +
"You are the entire ocean, in a drop."
```

Problem

What if we wanted to represent the " character in our string?



Can you think of a solution?

Problem

What if we wanted to represent the " character in our string?

Solution: \"

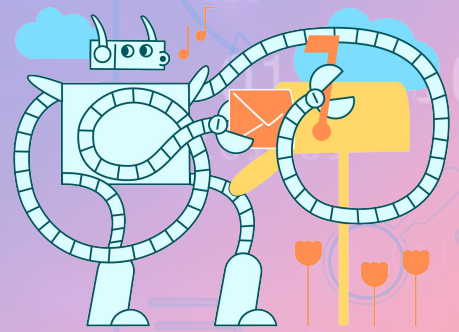
```
let myString = "\"hello\""
```

But what if we wanted to represent \?

Solution: \\

Bonus Self-Reading

Template Strings



```
let a = 1337
let myString = `The number is ${a}`
```

This is used to reference a variable within a string

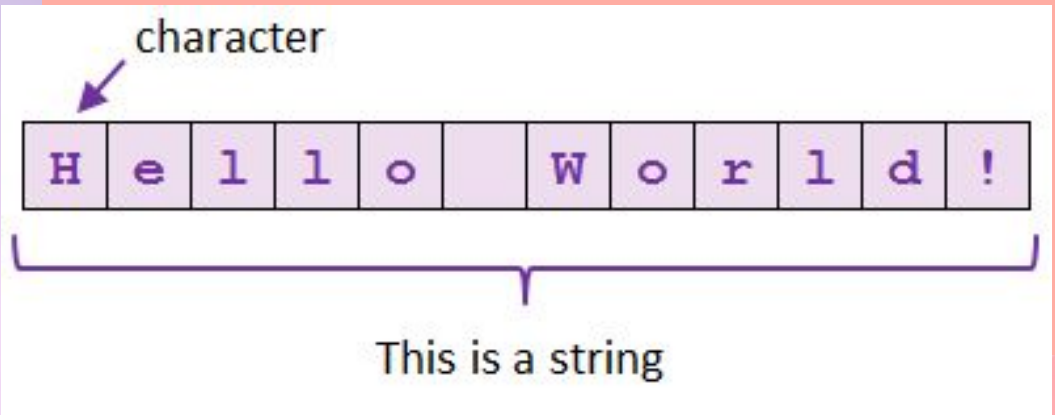
```
Console Output

> The number is 1337
```

String Methods

And properties

Remember?

Strings are just a **bunch of characters** "stringed" together in a series

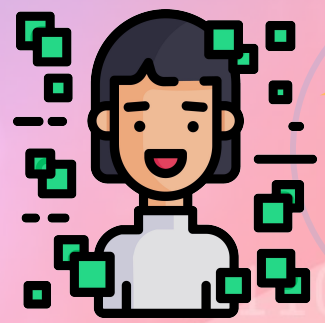


Strings

It's possible to get a character from the string by its **index**

Like saying "I want the third book on the shelf"

a	b	i	l	i	t	y



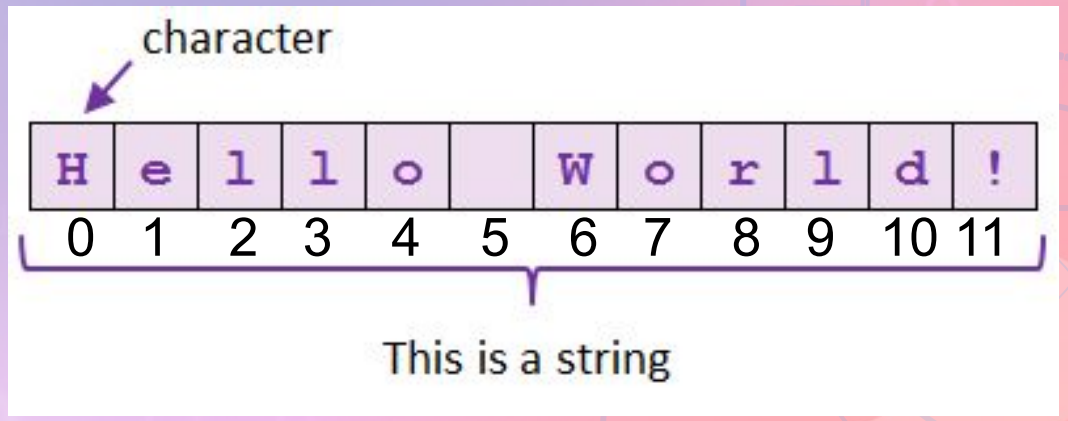
How do we get here?

Strings

Computers like to count from **0** instead of **1**

So to get the first character we can write

```
let myString = "my string"
let firstChar = myString[0]
```



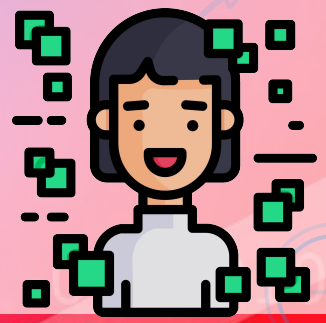
We put the “index” between square brackets

Strings

Therefore:

a	b	i	l	i	t	y

```
"ability"[2]
```

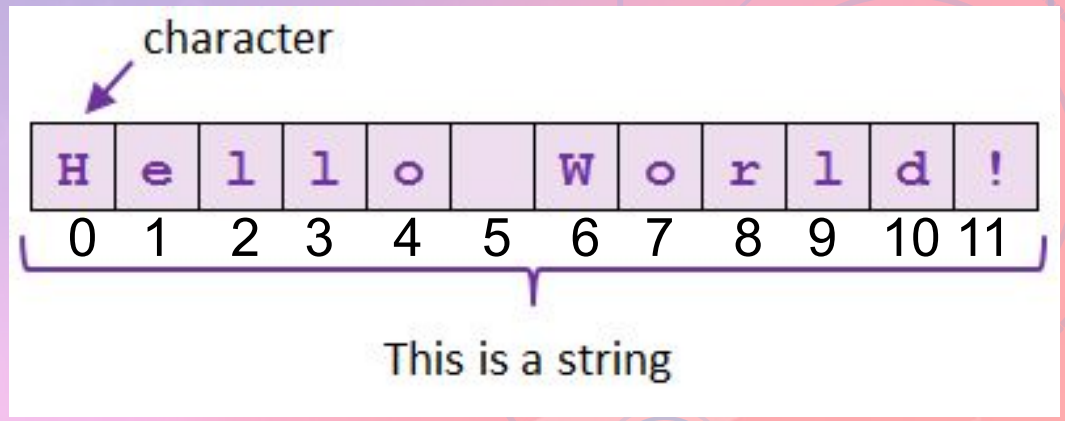


String Length

Every string has a **“length”** property with the number of characters in the string.

```
let myString = "Hello World!"
console.log(myString.length)
```

```
Console Output

> 12
```



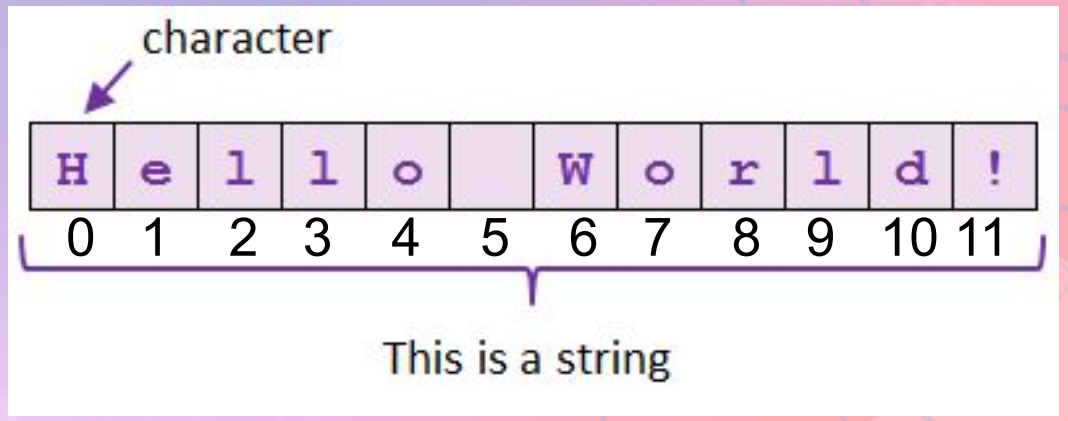
Question

How would you get the **last character** of a string?

```
string[string.length - 1]
```

Why -1?

0 based indexing!



Out of Range

If you try to access an **index >= string.length**

a	b	i	l	i	t	y	

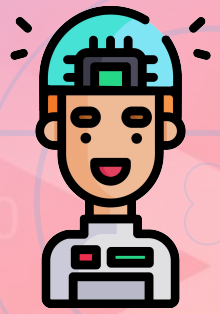
The result will be undefined

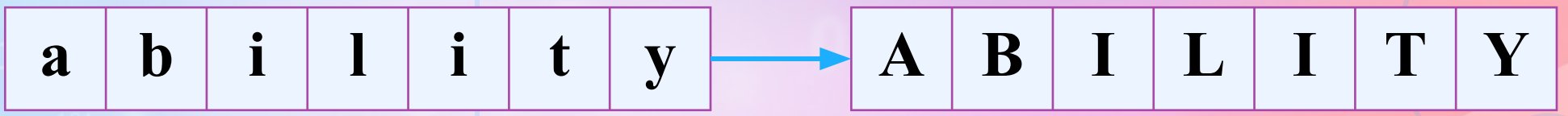
String Methods

Every string also has **“methods”**

These are operations that can be performed on the string instance

For example: changing the case of the string





String Case Conversion

Want to get your message across?

YELL IT IN ALL UPPERCASE LETTERS!

```
let myMessage = "Pineapple 1 dollar".toUpperCase()
```

PINEAPPLE 1 DOLLAR

String Case Conversion

Similarly we can use string.toLowerCase() to convert the string to all lower letters.

```
let myMessage = “PINEAPPLE 1 DOLLAR".toLowerCase()
```

PINEAPPLE 1 DOLLAR → pineapple 1 dollar

String Case Conversion

```
let myMessage = “PINEAPPLE 1 DOLLAR".toLowerCase()
```

Notice that we’re invoking (== performing) the “toLowerCase” operation on the string when we add the () parenthesis

Secret Code

Money = Tea

Dropoff = River

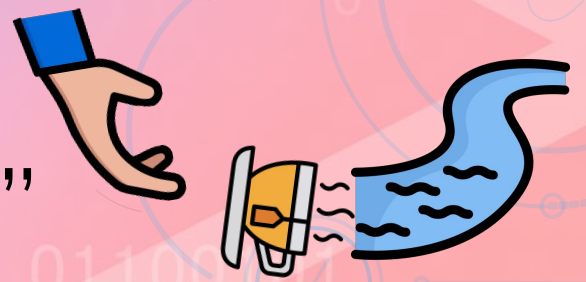
Send = Throw

“send the money to the drop-off tomorrow at 10”

Secret Code

```
let message  = "send the money to the drop-off tomorrow at 1"

message = message.replace("money", "tea")
message = message.replace("dropoff", "river")
message = message.replace("send", "throw")
```

“throw the tea to the river tomorrow at 10”



Slice

let cake = "▤▦▨▩▤▦▨▩"

Wow. Look at that delicious cake!

Let's cut a slice of it and eat it 😋

```
cake.slice(0, 2)
```

→ ▤▦

Slice

let cake = “▤▦▨▩▤▦▨▩”

Slice returns the characters between the starting index and the ending index (end not included)

```
cake.slice(2, 4)
```

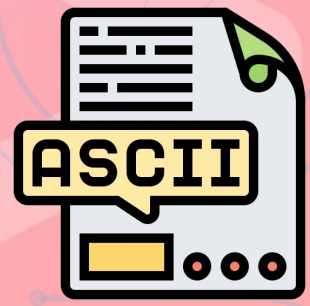
▨▩

ASCII

Using these numbers, we can represent all the different letters!

Character	Code	Binary
A	65	1000001
B	66	1000010
a	97	1100001
0	30	11110

This is called the **ASCII character encoding**



Character codes in JavaScript

Strings have the charCodeAt method:

```
"A".charCodeAt(0)
// 65
```

ASCII code for “A”: 65

And to convert a code to a string:

```
String.fromCharCode(65)
```

Summary

Summary

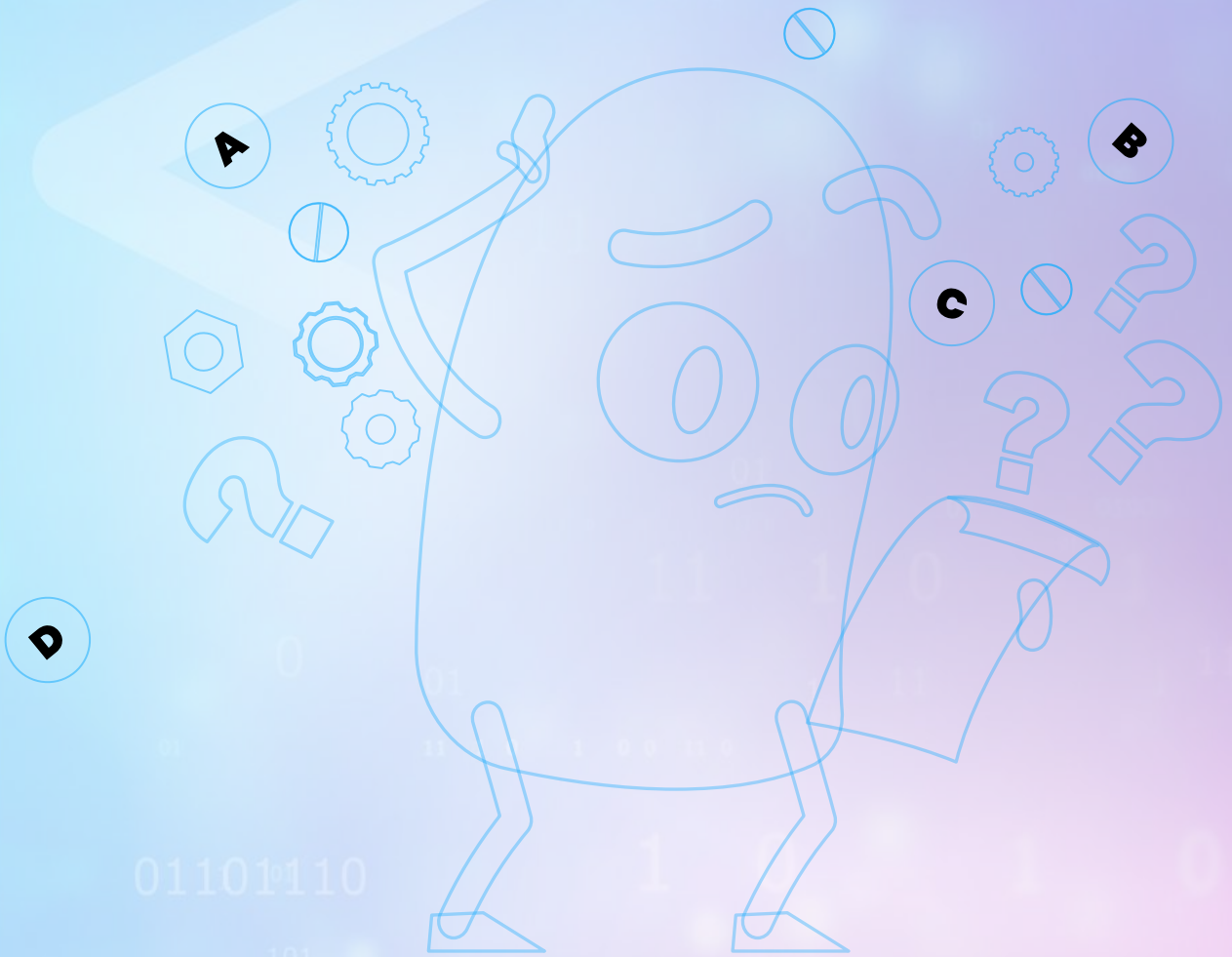
Newline:	`\n`
To represent “ character:	`let myString = "\"hello\""`
To get the char at a particular index	`let firstChar = myString[0]`
To get the length of a string	`myString.length`

Summary

Convert string to uppercase	`myString.toUpperCase()`
Convert string to lowercase	`myString.toLowerCase()`
Replace occurrences of a substring	`myString.replace("money", "tea")`
Returns chars between starting and ending index	`myString.slice(0, 2)`

Summary

Convert char to code	`"A".charCodeAt(0)` `// 65`
Convert code to char	`String.fromCharCode(65)`



Questions?

Sentinel

Your Turn!

> Play around, have fun, ask questions!